Máster Título Propio
Programación de Videojuegos

tech universidad ULAC

tech universidad ULAC

Máster Título Propio
Programación de Videojuegos

- Modalidad: **online**
- Duración: **12 meses**
- Titulación: **TECH Universidad ULAC**
- Acreditación: **60 ECTS**
- Horario: **a tu ritmo**
- Exámenes: **online**

Acceso web: www.techtitute.com/videojuegos/master/master-programacion-videojuegos

Índice

01

Presentación

De entre las tareas más importantes y delicadas a la hora de llevar a cabo el proyecto de un videojuego se encuentra la programación. La programación constituye el núcleo del videojuego, puesto que es el proceso que crea sus instrucciones básicas y dicta su funcionamiento general. Es decir, sin el código realizado por los desarrolladores el apartado visual, la historia y la jugabilidad no podrían destacar en una obra audiovisual de este tipo. Así, esta titulación ofrece a sus alumnos todos los conocimientos para convertirse en los mejores programadores de la industria, de forma que las mejores compañías quieran contar con ellos para desarrollar sus proyectos.

Aprende a programar los mejores videojuegos del mundo gracias a este Máster Título Propio"

La industria del videojuego ha experimentado una gran expansión en los últimos años. Con la popularización de esta forma de ocio, las compañías del sector se han visto obligadas a diseñar y publicar juegos con mayor frecuencia. Además, también se ha necesitado más creatividad debido a que los jugadores cada vez exigen títulos más variados, de diferentes géneros y que ofrezcan novedosas experiencias.

Por esa razón, este sector demanda especialistas en programación de videojuegos para que puedan encargarse de la tarea fundamental de crear el código de sus nuevas obras. Esta labor es delicada y requiere de una gran especialización, por lo que conviene haber llevado a cabo un proceso de aprendizaje profundo y óptimo para convertirse en un auténtico experto.

Así, este Máster Título Propio en Programación de Videojuegos es lo que los profesionales necesitan para poder acceder a una gran compañía de la industria trabajando en su departamento de desarrollo. A lo largo de esta titulación los alumnos podrán aprender fundamentos de programación e ingeniería de software, estructura de datos y algoritmos, programación orientada a objetos y otras cuestiones más específicas como los motores de videojuegos o la programación en tiempo real.

De esta forma, se garantiza que los estudiantes consigan los mejores conocimientos para que puedan aplicarlos directamente en sus ámbitos laborales.

Este **Máster Título Propio en Programación de Videojuegos** contiene el programa educativo más completo y actualizado del mercado. Las características más destacadas son:

- El desarrollo de casos prácticos presentados por expertos en programación y desarrollo de videojuegos
- Los contenidos gráficos, esquemáticos y eminentemente prácticos con los que está concebido recogen una información científica y práctica sobre aquellas disciplinas indispensables para el ejercicio profesional
- Los ejercicios prácticos donde realizar el proceso de autoevaluación para mejorar el aprendizaje
- Su especial hincapié en metodologías innovadoras
- Las lecciones teóricas, preguntas al experto, foros de discusión de temas controvertidos y trabajos de reflexión individual
- La disponibilidad de acceso a los contenidos desde cualquier dispositivo fijo o portátil con conexión a internet

Las mejores empresas del sector querrán contar contigo”

Quieres desarrollar los mejores videojuegos del mundo y esta titulación te enseña cómo conseguirlo”

Programa los videojuegos de tus sueños gracias a este Máster Título Propio.

No esperes más: programa videojuegos como los mejores expertos.

El programa incluye en su cuadro docente a profesionales del sector que vierten en esta capacitación la experiencia de su trabajo, además de reconocidos especialistas de sociedades de referencia y universidades de prestigio.

Su contenido multimedia, elaborado con la última tecnología educativa, permitirá al profesional un aprendizaje situado y contextual, es decir, un entorno simulado que proporcionará una capacitación inmersiva programada para entrenarse ante situaciones reales.

El diseño de este programa se centra en el Aprendizaje Basado en Problemas, mediante el cual el profesional deberá tratar de resolver las distintas situaciones de práctica profesional que se le planteen a lo largo del curso académico. Para ello, contará con la ayuda de un novedoso sistema de vídeo interactivo realizado por reconocidos expertos.

02

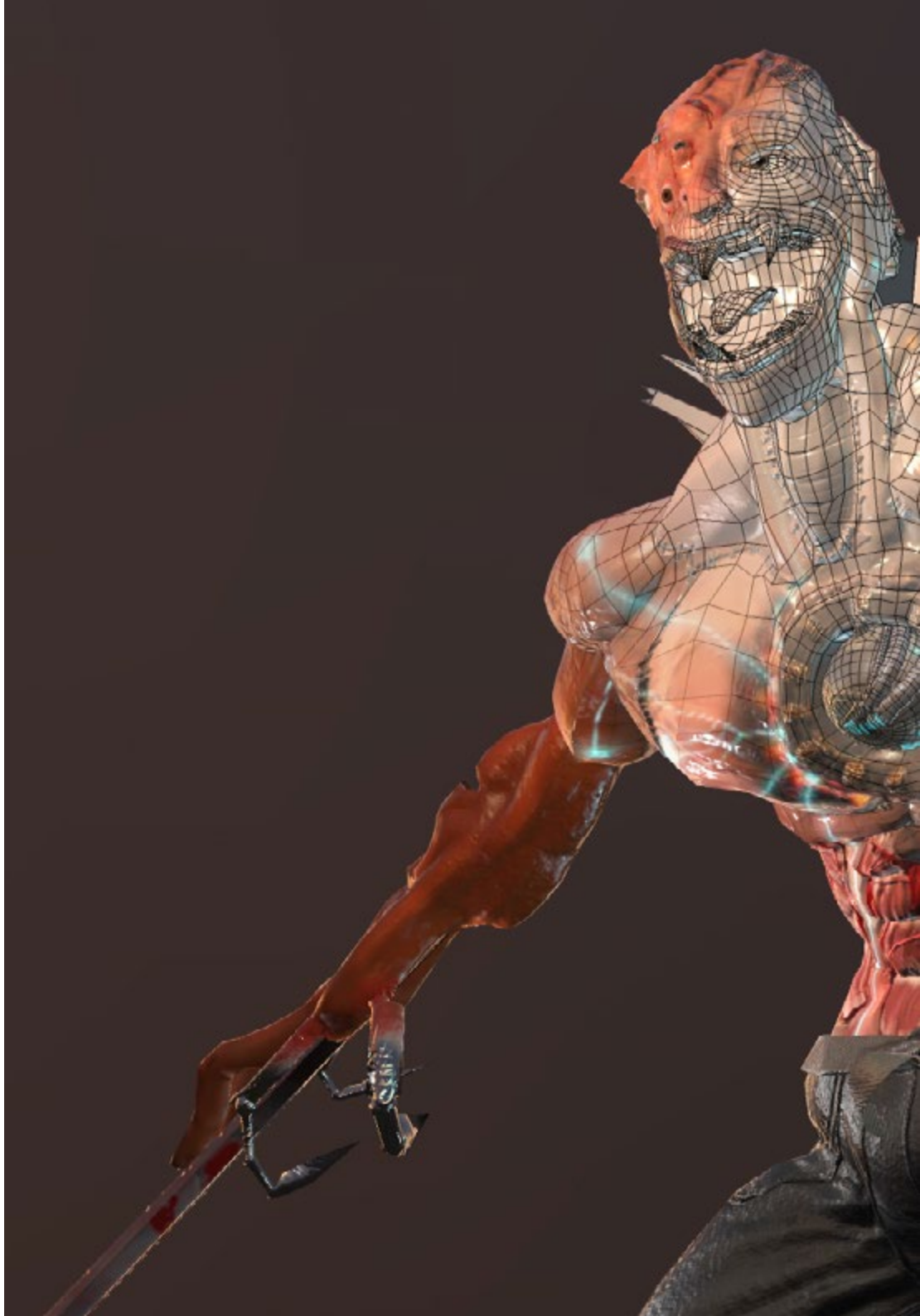
Objetivos

El objetivo principal de este Máster Título Propio en Programación de Videojuegos es ofrecer a sus alumnos los mejores conocimientos para que se conviertan en los mayores expertos en desarrollo de videojuegos de su entorno. Para ello, esta titulación les ofrece una serie de herramientas aplicadas a este ámbito que harán que su trabajo como desarrolladores mejore y los lleve a alcanzar todos sus objetivos profesionales, pudiendo programar los mejores videojuegos del mundo.

Consigue todos tus objetivos gracias a esta titulación”

Objetivos generales

- Conocer los diferentes lenguajes y métodos de programación aplicados al videojuego
- Profundizar en el proceso de producción de un videojuego y en la integración de la programación en estas etapas
- Aprender los fundamentos del diseño de videojuegos y aquellos conocimientos teóricos que un diseñador de videojuegos debe conocer
- Dominar los lenguajes de programación básicos empleados en videojuegos
- Aplicar conocimientos de la ingeniería de software y programación especializada a los videojuegos
- Entender el papel de la programación en el desarrollo de un videojuego
- Conocer las distintas consolas y plataformas existentes
- Desarrollar videojuegos web y multijugador



Objetivos específicos

Módulo 1. Fundamentos de programación

- Comprender la estructura básica de un ordenador, el software y de los lenguajes de programación de propósito general
- Analizar los elementos esenciales de un programa informático, como son los distintos tipos de datos, operadores, expresiones, sentencias, E/S y sentencias de control
- Interpretar algoritmos, que son la base necesaria para poder desarrollar programas informáticos

Módulo 2. Estructura de datos y algoritmos

- Aprender las principales estrategias de diseño de algoritmos, así como los distintos métodos y medidas para de cálculo de los mismos
- Distinguir el funcionamiento de los algoritmos, su estrategia y ejemplos de su uso en los principales problemas conocidos
- Entender la técnica de *Backtracking* y sus principales usos

Módulo 3. Programación orientada a objetos

- Conocer los distintos patrones de diseño para problemas orientados a objetos
- Entender la importancia de la documentación y las pruebas en el desarrollo del software
- Gestionar el uso de los hilos y la sincronización, así como la resolución de los problemas comunes dentro de la programación concurrente

Módulo 4. Consolas y dispositivos para videojuegos

- Saber el funcionamiento básico de los principales periféricos de entrada y salida
- Entender las principales implicaciones de diseño de las diferentes plataformas
- Estudiar la estructura, organización, funcionamiento e interconexión de los dispositivos y sistemas
- Comprender la función del sistema operativo y los kits de desarrollo para dispositivos móviles y plataformas de videojuegos

Módulo 5. Ingeniería de software

- Distinguir las bases de la ingeniería del software, así como el proceso del software y los distintos modelos para su desarrollo incluyendo tecnologías ágiles
- Reconocer la ingeniería de requisitos, su desarrollo, elaboración, negociación y validación a fin de entender las principales normas relativas a la calidad del software y a la administración de proyectos

Módulo 6. Motores de videojuegos

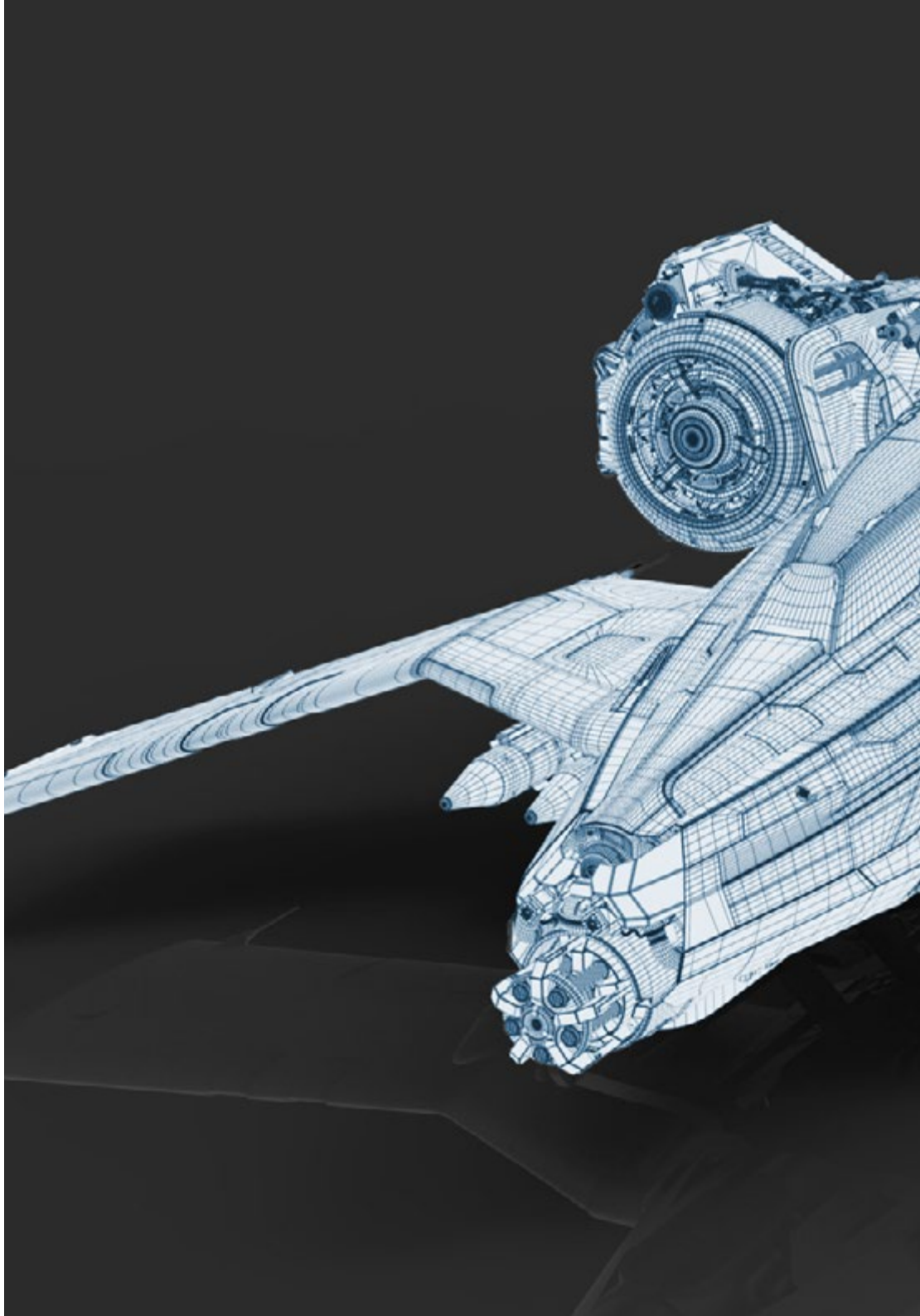
- Descubrir el funcionamiento y la arquitectura de un motor de videojuegos
- Comprender las características básicas de los motores de juegos existentes
- Programar aplicaciones de manera correcta y eficiente aplicadas a motores de videojuegos
- Elegir el paradigma y los lenguajes de programación más apropiados para programar aplicaciones aplicadas a motores de videojuegos

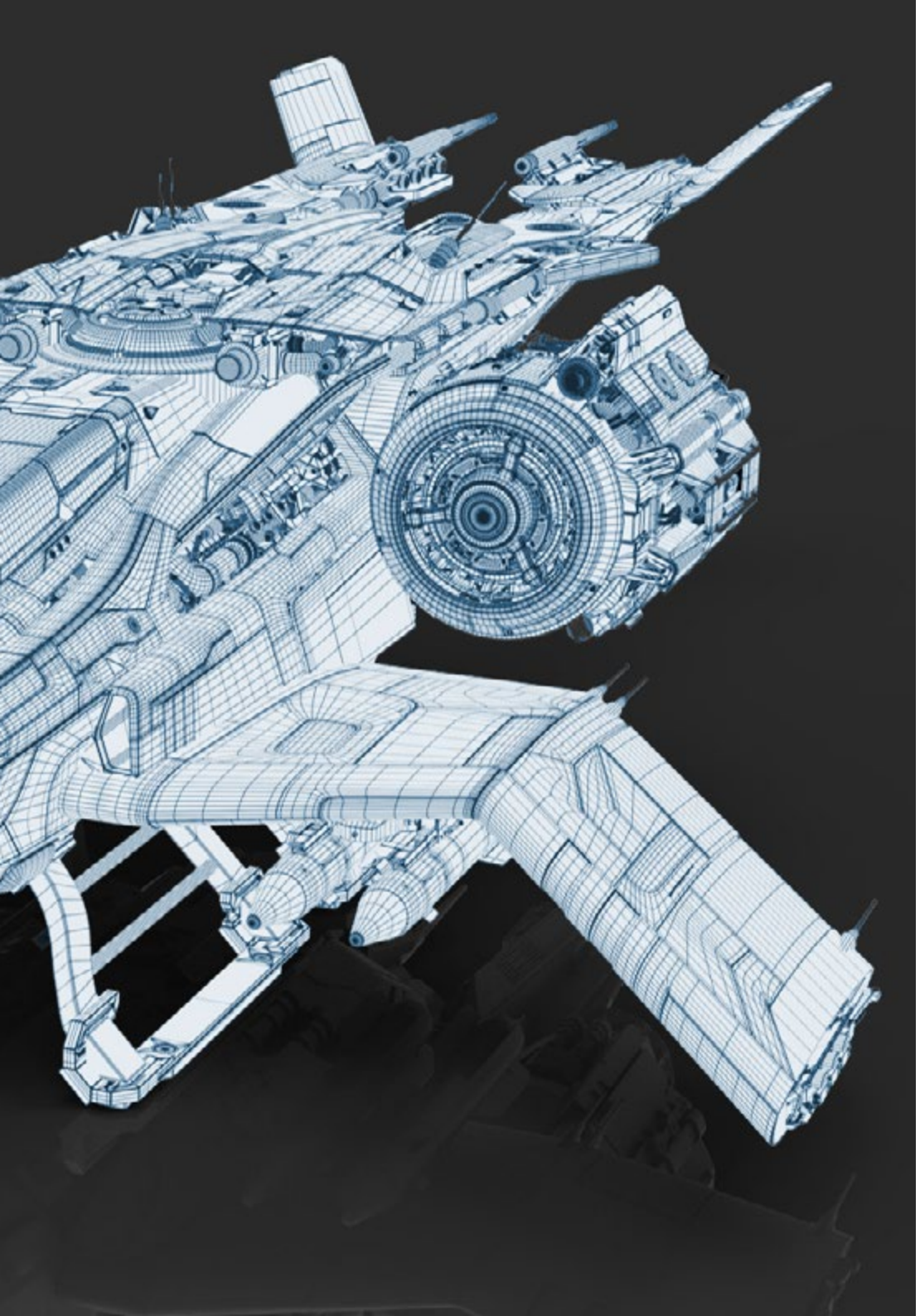
Módulo 7. Sistemas inteligentes

- Establecer los conceptos relacionados con la teoría de agentes y la arquitectura de agentes y su proceso de razonamiento
- Asimilar la teoría y la práctica detrás de los conceptos de información y conocimiento, así como las distintas maneras de representar el conocimiento
- Comprender el funcionamiento de los razonadores semánticos, los sistemas basados en conocimiento y los sistemas expertos

Módulo 8. Programación en tiempo real

- Analizar las características clave de un lenguaje de programación en tiempo real que lo diferencian del lenguaje de programación tradicional
- Comprender los conceptos básicos de los sistemas informáticos
- Adquirir la capacidad de aplicar las principales bases y técnicas de programación en tiempo real





Módulo 9. Diseño y desarrollo de juegos web

- Diseñar juegos y aplicaciones web interactivas con la documentación correspondiente
- Evaluar las características principales de los juegos y las aplicaciones web interactivas para comunicarse de manera profesional y correcta

Módulo 10. Redes y sistemas multijugador

- Describir la arquitectura del protocolo de control de transmisión/protocolo de Internet (TCP / IP) y el funcionamiento básico de las redes inalámbricas
- Analizando la seguridad aplicada a videojuegos
- Adquirir la capacidad para desarrollar juegos en línea para múltiples jugadores

Quieres conseguir un puesto en las mejores compañías del mundo y esta titulación te ayudará a lograrlo"

03

Competencias

Los alumnos de este Máster Título Propio obtendrán una serie de habilidades que les convertirán en auténticos expertos en desarrollo de videojuegos, de forma que puedan incorporarse a cualquier tipo de proyecto de la industria. Así, los estudiantes dominarán cuestiones relacionadas con diferentes lenguajes de programación específicos que se emplean en este tipo de productos audiovisuales, así como capacitaciones transversales que deben conocer como el ámbito de las consolas y plataformas y los motores de videojuegos.

Lo sabrás todo para desarrollar grandes videojuegos”

Competencias generales

- Diseñar todas las fases de un videojuego, desde la idea inicial hasta el lanzamiento final
- Especializarse como programador de videojuegos
- Profundizar en todas las partes del desarrollo, desde la arquitectura inicial, la programación del personaje jugador y de todos los elementos que intervienen en el proceso de juego
- Obtener una visión de conjunto del proyecto, pudiendo aportar soluciones a las diferentes problemáticas y retos que surjan en el diseño de un videojuego

Domina toda clase de lenguajes de programación aplicados a los videojuegos con este Máster Título Propio"

Competencias específicas

- Conocer el software necesario para ser un profesional del desarrollo de videojuegos
- Comprender la experiencia del jugador y saber analizar la jugabilidad del videojuego
- Entender todo el procedimiento teórico y práctico del proceso de programación de un videojuego
- Dominar los lenguajes de programación más útiles para el mundo del videojuego
- Integrar la programación aprendida a diferentes tipos de consolas y plataformas
- Programar videojuegos web y multijugador
- Asimilar el concepto de motor de videojuegos para poder programar de forma correcta
- Aplicar conocimientos de ingeniería de software a la programación de videojuegos

04

Estructura y contenido

Este Máster Título Propio en Programación de Videojuegos ofrece a sus alumnos los mejores contenidos en desarrollo de videojuegos, gracias a su cuidadoso diseño, estructurado en 10 módulos de 10 temas cada uno. A través de ellos los estudiantes podrán aprender todo lo necesario para participar en cualquier clase de proyecto de videojuego, de forma que su proceso educativo sea completo, integral y totalmente enfocado a la práctica.

Los contenidos que necesitas para especializarte en programación de videojuegos”

Módulo 1. Fundamentos de programación

1.1. Introducción a la programación
- 1.1.1. Estructura básica de un ordenador
- 1.1.2. Software
- 1.1.3. Lenguajes de programación
- 1.1.4. Ciclo de vida de una aplicación informática

1.2. Diseño de algoritmos
- 1.2.1. La resolución de problemas
- 1.2.2. Técnicas deScriptivas
- 1.2.3. Elementos y estructura de un algoritmo

1.3. Elementos de un programa
- 1.3.1. Origen y características del lenguaje C++
- 1.3.2. El entorno de desarrollo
- 1.3.3. Concepto de programa
- 1.3.4. Tipos de datos fundamentales
- 1.3.5. Operadores
- 1.3.6. Expresiones
- 1.3.7. Sentencias
- 1.3.8. Entrada y salida de datos

1.4. Sentencias de control
- 1.4.1. Sentencias
- 1.4.2. Bifurcaciones
- 1.4.3. Bucles

1.5. Abstracción y modularidad: funciones
- 1.5.1. Diseño modular
- 1.5.2. Concepto de función y utilidad
- 1.5.3. Definición de una función
- 1.5.4. Flujo de ejecución en la llamada de una función
- 1.5.5. Prototipo de una función
- 1.5.6. Devolución de resultados
- 1.5.7. Llamada a una función: parámetros
- 1.5.8. Paso de parámetros por referencia y por valor
- 1.5.9. Ámbito identificador

1.6. Estructuras de datos estáticas
- 1.6.1. *Arrays*
- 1.6.2. Matrices. Poliedros
- 1.6.3. Búsqueda y ordenación
- 1.6.4. Cadenas. Funciones de E/S para cadenas
- 1.6.5. Estructuras. Uniones
- 1.6.6. Nuevos tipos de datos

1.7. Estructuras de datos dinámicas: punteros
- 1.7.1. Concepto. Definición de puntero
- 1.7.2. Operadores y operaciones con punteros
- 1.7.3. *Arrays* de punteros
- 1.7.4. Punteros y *arrays*
- 1.7.5. Punteros a cadenas
- 1.7.6. Punteros a estructuras
- 1.7.7. Indirección múltiple
- 1.7.8. Punteros a funciones
- 1.7.9. Paso de funciones, estructuras y *arrays* como parámetros de funciones

1.8. Ficheros
- 1.8.1. Conceptos básicos
- 1.8.2. Operaciones con ficheros
- 1.8.3. Tipos de ficheros
- 1.8.4. Organización de los ficheros
- 1.8.5. Introducción a los ficheros C++
- 1.8.6. Manejo de ficheros

1.9. Recursividad
- 1.9.1. Definición de recursividad
- 1.9.2. Tipos de recursión
- 1.9.3. Ventajas e inconvenientes
- 1.9.4. Consideraciones
- 1.9.5. Conversión recursivo-iterativa
- 1.9.6. La pila de recursión

1.10. Prueba y documentación
- 1.10.1. Pruebas de programas
- 1.10.2. Prueba de la caja blanca
- 1.10.3. Prueba de la caja negra
- 1.10.4. Herramientas para realizar las pruebas
- 1.10.5. Documentación de programas

Módulo 2. Estructura de datos y algoritmos

2.1. Introducción a las estrategias de diseño de algoritmos
- 2.1.1. Recursividad
- 2.1.2. Divide y conquista
- 2.1.3. Otras estrategias

2.2. Eficiencia y análisis de los algoritmos
- 2.2.1. Medidas de eficiencia
- 2.2.2. Medir el tamaño de la entrada
- 2.2.3. Medir el tiempo de ejecución
- 2.2.4. Caso peor, mejor y medio
- 2.2.5. Notación asintónica
- 2.2.6. Criterios de análisis matemático de algoritmos no recursivos
- 2.2.7. Análisis matemático de algoritmos recursivos
- 2.2.8. Análisis empírico de algoritmos

2.3. Algoritmos de ordenación
- 2.3.1. Concepto de ordenación
- 2.3.2. Ordenación de la burbuja
- 2.3.3. Ordenación por selección
- 2.3.4. Ordenación por inserción
- 2.3.5. Ordenación por mezcla (*Merge_Sort*)
- 2.3.6. Ordenación rápida (*Quick_Sort*)

2.4. Algoritmos con árboles
- 2.4.1. Concepto de árbol
- 2.4.2. Árboles binarios
- 2.4.3. Recorridos de árbol
- 2.4.4. Representar expresiones
- 2.4.5. Árboles binarios ordenados
- 2.4.6. Árboles binarios balanceados

2.5. Algoritmos con heaps
- 2.5.1. Los heaps
- 2.5.2. El algoritmo heapsort
- 2.5.3. Las colas de prioridad

2.6. Algoritmos con grafos
- 2.6.1. Representación
- 2.6.2. Recorrido en anchura
- 2.6.3. Recorrido en profundidd
- 2.6.4. Ordenación topológica

2.7. Algoritmos greedy
- 2.7.1. La estrategia greedy
- 2.7.2. Elementos de la estrategia greedy
- 2.7.3. Cambio de monedas
- 2.7.4. Problema del viajante
- 2.7.5. Problema de la mochila

2.8. Búsqueda de caminos mínimos
- 2.8.1. El problema del camino mínimo
- 2.8.2. Arcos negativos y ciclos
- 2.8.3. Algoritmo de Dijkstra

2.9. Algoritmos greedy sobre grafos
- 2.9.1. El árbol de recubrimiento mínimo
- 2.9.2. El algoritmo de Prim
- 2.9.3. El algoritmo de Kruskal
- 2.9.4. Análisis de complejidad

2.10. Backtracking
- 2.10.1. El *Backtracking*
- 2.10.2. Técnicas alternativas

Módulo 3. Programación orientada a objetos

3.1. Introducción a la programación orientada a objetos
- 3.1.1. Introducción a la programación orientada a objetos
- 3.1.2. Diseño de clases
- 3.1.3. Introducción a UML para el modelado de los problemas

3.2. Relaciones entre clases
- 3.2.1. Abstracción y herencia
- 3.2.2. Conceptos avanzados de herencia
- 3.2.3. Polimorfismo
- 3.2.4. Composición y agregación

3.3. Introducción a los patrones de diseño para problemas orientados a objetos
- 3.3.1. ¿Qué son los patrones de diseño?
- 3.3.2. Patrón *Factory*
- 3.3.3. Patrón *Singleton*
- 3.3.4. Patrón *Observer*
- 3.3.5. Patrón *Composite*

3.4. Excepciones
- 3.4.1. ¿Qué son las excepciones?
- 3.4.2. Captura y gestión de excepciones
- 3.4.3. Lanzamiento de excepciones
- 3.4.4. Creación de excepciones

3.5. Interfaces de usuarios
- 3.5.1. Introducción a Qt
- 3.5.2. Posicionamiento
- 3.5.3. ¿Qué son los eventos?
- 3.5.4. Eventos: definición y captura
- 3.5.5. Desarrollo de interfaces de usuario

3.6. Introducción a la programación concurrente
- 3.6.1. Introducción a la programación concurrente
- 3.6.2. El concepto de proceso e hilo
- 3.6.3. Interacción entre procesos o hilos
- 3.6.4. Los hilos en C++
- 3.6.5. Ventajas e inconvenientes de la programación concurrente

3.7. Gestión de hilos y sincronización
- 3.7.1. Ciclo de vida de un hilo
- 3.7.2. La clase *Thread*
- 3.7.3. Planificación de hilos
- 3.7.4. Grupos hilos
- 3.7.5. Hilos de tipo demonio
- 3.7.6. Sincronización
- 3.7.7. Mecanismos de bloqueo
- 3.7.8. Mecanismos de comunicación
- 3.7.9. Monitores

3.8. Problemas comunes dentro de la programación concurrente
- 3.8.1. El problema de los productores consumidores
- 3.8.2. El problema de los lectores y escritores
- 3.8.3. El problema de la cena de los filósofos

3.9. Documentación y pruebas de software
- 3.9.1. ¿Por qué es importante documentar el software?
- 3.9.2. Documentación de diseño
- 3.9.3. Uso de herramientas para la documentación

3.10. Pruebas de software
- 3.10.1. Introducción a las pruebas del software
- 3.10.2. Tipos de pruebas
- 3.10.3. Prueba de unidad
- 3.10.4. Prueba de integración
- 3.10.5. Prueba de validación
- 3.10.6. Prueba del sistema

Módulo 4. Consolas y dispositivos para videojuegos

4.1. Historia de la programación en videojuegos
- 4.1.1. Periodo Atari (1977-1985)
- 4.1.2. Periodo NES y SNES (1985-1995)
- 4.1.3. Periodo PlayStation / PlayStation 2 (1995-2005)
- 4.1.4. Periodo Xbox 360, PS3 y Wii (2005-2013)
- 4.1.5. Periodo Xbox One, PS4 y Wii U – Switch (2013-actualidad)
- 4.1.6. El futuro

4.2. Historia de la jugabilidad en videojuegos
- 4.2.1. Introducción
- 4.2.2. Contexto social
- 4.2.3. Diagrama estructural
- 4.2.4. Futuro

4.3. Adaptación a los tiempos modernos
- 4.3.1. Juegos basados en movimiento
- 4.3.2. Realidad virtual
- 4.3.3. Realidad aumentada
- 4.3.4. Realidad mixta

4.4. *Unity: Scripting I* y ejemplos
- 4.4.1. ¿Qué es un Script?
- 4.4.2. Nuestro primer Script
- 4.4.3. Añadiendo un Script
- 4.4.4. Abriendo un Script
- 4.4.5. MonoBehaviour
- 4.4.6. *Debugging*

4.5. *Unity: Scripting II* y ejemplos
- 4.5.1. Entrada de teclado y ratón
- 4.5.2. Raycast
- 4.5.3. Instanciación
- 4.5.4. Variables
- 4.5.5. Variables públicas y serializadas

4.6. *Unity: Scripting III* y ejemplos
- 4.6.1. Obteniendo componentes
- 4.6.2. Modificando componentes
- 4.6.3. Testeo
- 4.6.4. Múltiples objetos
- 4.6.5. *Colliders y triggers*
- 4.6.6. Cuaterniones

4.7. Periféricos
- 4.7.1. Evolución y clasificación
- 4.7.2. Periféricos e interfaces
- 4.7.3. Periféricos actuales
- 4.7.4. Futuro próximo

4.8. Videojuegos: perspectivas futuras
- 4.8.1. Juego basado en la nube
- 4.8.2. Ausencia de controladores
- 4.8.3. Realidad inmersiva
- 4.8.4. Otras alternativas

4.9. Arquitectura
- 4.9.1. Necesidades especiales de los videojuegos
- 4.9.2. Evolución de la arquitectura
- 4.9.3. Arquitectura actual
- 4.9.4. Diferencias entre arquitecturas

4.10. Kits de desarrollo y su evolución
- 4.10.1. Introducción
- 4.10.2. Tercera generación de kits de desarrollo
- 4.10.3. Cuarta generación de kits de desarrollo
- 4.10.4. Quinta generación de kits de desarrollo
- 4.10.5. Sexta generación de kits de desarrollo

Módulo 5. Ingeniería de software

5.1. Introducción a la ingeniería del software y al modelado
- 5.1.1. La naturaleza del software
- 5.1.2. La naturaleza única de las webapps
- 5.1.3. Ingeniería del software
- 5.1.4. El proceso del software
- 5.1.5. La práctica de la ingeniería del software
- 5.1.6. Mitos del software
- 5.1.7. ¿Cómo comienza todo?
- 5.1.8. Conceptos orientados a objetos
- 5.1.9. Introducción a UML

5.2. El proceso del software
- 5.2.1. Un modelo general de proceso
- 5.2.2. Modelos de proceso preScriptivos
- 5.2.3. Modelos de proceso especializado
- 5.2.4. El proceso unificado
- 5.2.5. Modelos del proceso personal y del equipo
- 5.2.6. ¿Qué es la agilidad?
- 5.2.7. ¿Qué es un proceso ágil?
- 5.2.8. Scrum
- 5.2.9. Conjunto de herramientas para el proceso ágil

5.3. Principios que guían la práctica de la ingeniería del software
- 5.3.1. Principios que guían el proceso
- 5.3.2. Principios que guían la práctica
- 5.3.3. Principios de comunicación
- 5.3.4. Principios de planificación
- 5.3.5. Principios de modelado
- 5.3.6. Principios de construcción
- 5.3.7. Principios de despliegue

5.4. Comprensión de los requisitos
- 5.4.1. Ingeniería de requisitos
- 5.4.2. Establecer las bases
- 5.4.3. Indagación de los requisitos
- 5.4.4. Desarrollo de casos de uso
- 5.4.5. Elaboración del modelo de los requisitos
- 5.4.6. Negociación de los requisitos
- 5.4.7. Validación de los requisitos

5.5. Modelado de los requisitos: escenarios, información y clases de análisis
- 5.5.1. Análisis de los requisitos
- 5.5.2. Modelado basado en escenarios
- 5.5.3. Modelos UML que proporcionan el caso de uso
- 5.5.4. Conceptos de modelado de datos
- 5.5.5. Modelado basado en clases
- 5.5.6. Diagramas de clases

5.6. Modelado de los requisitos: flujo, comportamiento y patrones
- 5.6.1. Requisitos que modelan las estrategias
- 5.6.2. Modelado orientado al flujo
- 5.6.3. Diagramas de estado
- 5.6.4. Creación de un modelo de comportamiento
- 5.6.5. Diagramas de secuencia
- 5.6.6. Diagramas de comunicación
- 5.6.7. Patrones para el modelado de requisitos

5.7. Conceptos de diseño
- 5.7.1. Diseño en el contexto de la ingeniería del software
- 5.7.2. El proceso de diseño
- 5.7.3. Conceptos de diseño
- 5.7.4. Conceptos de diseño orientado a objetos
- 5.7.5. El modelo del diseño

5.8. Diseño de la arquitectura
- 5.8.1. Arquitectura del software
- 5.8.2. Géneros arquitectónicos
- 5.8.3. Estilos arquitectónicos
- 5.8.4. Diseño arquitectónico
- 5.8.5. Evolución de los diseños alternativos para la arquitectura
- 5.8.6. Mapeo de la arquitectura con el uso del flujo de datos

5.9. Diseño en el nivel de componentes y basado en patrones
- 5.9.1. ¿Qué es un componente?
- 5.9.2. Diseño de componentes basados en clase
- 5.9.3. Realización del diseño en el nivel de componentes
- 5.9.4. Diseño de componentes tradicionales
- 5.9.5. Desarrollo basado en componentes
- 5.9.6. Patrones de diseño
- 5.9.7. Diseño de software basado en patrones
- 5.9.8. Patrones arquitectónicos
- 5.9.9. Patrones de diseño en el nivel de componentes
- 5.9.10. Patrones de diseño de la interfaz de usuario

5.10. Calidad del software y administración de proyectos
- 5.10.1. Calidad
- 5.10.2. Calidad del software
- 5.10.3. El dilema de la calidad del software
- 5.10.4. Lograr la calidad del software
- 5.10.5. Aseguramiento de la calidad del software
- 5.10.6. El espectro administrativo
- 5.10.7. El personal
- 5.10.8. El producto
- 5.10.9. El proceso
- 5.10.10. El proyecto
- 5.10.11. Principios y prácticas

Módulo 6. Motores de videojuegos

6.1. Los videojuegos y las TIC
- 6.1.1. Introducción
- 6.1.2. Oportunidades
- 6.1.3. Desafíos
- 6.1.4. Conclusiones

6.2. Historia de los motores de videojuegos
- 6.2.1. Introducción
- 6.2.2. Época Atari
- 6.2.3. Época de los 80
- 6.2.4. Primeros motores. Época de los 90
- 6.2.5. Motores actuales

6.3. Motores de videojuegos
- 6.3.1. Tipos de motores
- 6.3.2. Partes de un motor de videojuegos
- 6.3.3. Motores actuales
- 6.3.4. Selección de un motor para nuestro proyecto

6.4. *Motor Game Maker*
- 6.4.1. Introducción
- 6.4.2. Diseño de escenarios
- 6.4.3. *Sprites* y animaciones
- 6.4.4. Colisiones
- 6.4.5. Scripting en GML

6.5. Motor Unreal Engine 4: introducción
- 6.5.1. ¿Qué es Unreal Engine 4? ¿Cuál es su filosofía?
- 6.5.2. Materiales
- 6.5.3. UI
- 6.5.4. Animaciones
- 6.5.5. Sistema de partículas
- 6.5.6. Inteligencia artificial
- 6.5.7. FPS

6.6. Motor Unreal Engine 4: Visual Scripting
- 6.6.1. Filosofía de los *Blueprints* y el *Visual Scripting*
- 6.6.2. *Debugging*
- 6.6.3. Tipos de variables
- 6.6.4. Control de flujo básico

6.7. Motor Unity 5
- 6.7.1. Programación en C# y Visual Studio
- 6.7.2. Creación de *Prefabs*
- 6.7.3. Uso de Gizmos para el control del videojuego
- 6.7.4. Motor adaptativo: 2D y 3D

6.8. Motor Godot
- 6.8.1. Filosofía de diseño de Godot
- 6.8.2. Diseño orientado a objetos y composición
- 6.8.3. Todo incluido en un paquete
- 6.8.4. Software libre y dirigido por la comunidad

6.9. Motor RPG Maker
- 6.9.1. Filosofía de RPG Maker
- 6.9.2. Tomando como referencia
- 6.9.3. Crear un juego con personalidad
- 6.9.4. Juegos comerciales de éxito

6.10. Motor Source 2
- 6.10.1. Filosofía de Source 2
- 6.10.2. Source y Source 2: evolución
- 6.10.3. Uso de la comunidad: contenido audiovisual y videojuegos
- 6.10.4. Futuro del motor Source 2
- 6.10.5. Mods y juegos de éxito

Módulo 7. Sistemas inteligentes

7.1. Teoría de agentes
- 7.1.1. Historia del concepto
- 7.1.2. Definición de agente
- 7.1.3. Agentes en inteligencia artificial
- 7.1.4. Agentes en Ingeniería de Software

7.2. Arquitecturas de agentes
- 7.2.1. El proceso de razonamiento de un agente
- 7.2.2. Agentes reactivos
- 7.2.3. Agentes deductivos
- 7.2.4. Agentes híbridos
- 7.2.5. Comparativa

7.3. Información y conocimiento
- 7.3.1. Distinción entre datos, información y conocimiento
- 7.3.2. Evaluación de la calidad de los datos
- 7.3.3. Métodos de captura de datos
- 7.3.4. Métodos de adquisición de información
- 7.3.5. Métodos de adquisición de conocimiento

7.4. Representación del conocimiento
- 7.4.1. La importancia de la representación del conocimiento
- 7.4.2. Definición de representación del conocimiento a través de sus roles
- 7.4.3. Características de una representación del conocimiento

7.5. Ontologías
- 7.5.1. Introducción a los metadatos
- 7.5.2. Concepto filosófico de ontología
- 7.5.3. Concepto informático de ontología
- 7.5.4. Ontologías de dominio y ontologías de nivel superior
- 7.5.5. ¿Cómo construir una ontología?

7.6. Lenguajes para ontologías y software para la creación de ontologías
- 7.6.1. Tripletas RDF, Turtle y N3
- 7.6.2. RDF Schema
- 7.6.3. OWL
- 7.6.4. SPARQL
- 7.6.5. Introducción a las diferentes herramientas para la creación de ontologías
- 7.6.6. Instalación y uso de Protégé

7.7. La web semántica
- 7.7.1. El estado actual y futuro de la web semántica
- 7.7.2. Aplicaciones de la web semántica

7.8. Otros modelos de representación del conocimiento
- 7.8.1. Vocabularios
- 7.8.2. Visión global
- 7.8.3. Taxonomías
- 7.8.4. Tesauros
- 7.8.5. Folksonomías
- 7.8.6. Comparativa
- 7.8.7. Mapas mentales

7.9. Evaluación e integración de representaciones del conocimiento
- 7.9.1. Lógica de orden cero
- 7.9.2. Lógica de primer orden
- 7.9.3. Lógica deScriptiva
- 7.9.4. Relación entre diferentes tipos de lógica
- 7.9.5. Prolog: programación basada en lógica de primer orden

7.10. Razonadores semánticos, sistemas basados en conocimiento y sistemas expertos
- 7.10.1. Concepto de razonador
- 7.10.2. Aplicaciones de un razonador
- 7.10.3. Sistemas basados en el conocimiento
- 7.10.4. MYCIN, historia de los Sistemas Expertos
- 7.10.5. Elementos y Arquitectura de Sistemas Expertos
- 7.10.6. Creación de Sistemas Expertos

Módulo 8. Programación en tiempo real

8.1. Conceptos básicos de la programación concurrente
- 8.1.1. Conceptos fundamentales
- 8.1.2. Concurrencia
- 8.1.3. Beneficios de la concurrencia
- 8.1.4. Concurrencia y hardware

8.2. Estructuras básicas de soporte a la concurrencia en Java
- 8.2.1. Concurrencia en Java
- 8.2.2. Creación de *Threads*
- 8.2.3. Métodos
- 8.2.4. Sincronización

8.3. *Threads*, ciclo de vida, prioridades, interrupciones, estados, ejecutores
- 8.3.1. *Threads*
- 8.3.2. Ciclo de vida
- 8.3.3. Prioridades
- 8.3.4. Interrupciones
- 8.3.5. Estados
- 8.3.6. Ejecutores

8.4. Exclusión mutua
- 8.4.1. ¿Qué es la exclusión mutua?
- 8.4.2. Algoritmo de Dekker
- 8.4.3. Algoritmo de Peterson
- 8.4.4. Exclusión mutua en Java

8.5. Dependencias de estados
- 8.5.1. Inyección de dependencias
- 8.5.2. Implementación del patrón en Java
- 8.5.3. Formas de inyectar las dependencias
- 8.5.4. Ejemplo

8.6. Patrones de diseño
- 8.6.1. Introducción
- 8.6.2. Patrones de creación
- 8.6.3. Patrones de estructura
- 8.6.4. Patrones de comportamiento

8.7. Uso de bibliotecas Java
- 8.7.1. ¿Qué son las bibliotecas en Java?
- 8.7.2. *Mockito-All*, *Mockito-Core*
- 8.7.3. Guava
- 8.7.4. Commons-io
- 8.7.5. Commons-lang, commons-lang3

8.8. Programación de *Shaders*
- 8.8.1. Pipeline 3D y rasterizado
- 8.8.2. Vertex Shading
- 8.8.3. Pixel *Shading: Iluminación I*
- 8.8.4. Pixel *Shading: Iluminación II*
- 8.8.5. Post-effectos

8.9. Programación de tiempo real
- 8.9.1. Introducción
- 8.9.2. Procesamiento de interrupciones
- 8.9.3. Sincronización y comunicación entre procesos
- 8.9.4. Los sistemas de planificación en tiempo real

8.10. Planificación de tiempo real
- 8.10.1. Conceptos
- 8.10.2. Modelo de referencia de los sistemas de tiempo real
- 8.10.3. Políticas de planificación
- 8.10.4. Planificadores cíclicos
- 8.10.5. Planificadores con propiedades estáticas
- 8.10.6. Planificadores con propiedades dinámicas

Módulo 9. Diseño y desarrollo de juegos web

9.1. Orígenes y estándares de la web
- 9.1.1. Orígenes de Internet
- 9.1.2. Creación de *World Wide Web*
- 9.1.3. Aparición de los estándares web
- 9.1.4. El auge de los estándares web

9.2. HTTP y estructura cliente-servidor
- 9.2.1. Rol cliente-servidor
- 9.2.2. Comunicación cliente-servidor
- 9.2.3. Historia reciente
- 9.2.4. Computación centralizada

9.3. Programación web: introducción
- 9.3.1. Conceptos básicos
- 9.3.2. Preparando un servidor web
- 9.3.3. Conceptos básicos de HTML5
- 9.3.4. Formas HTML

9.4. Introducción a HTML y ejemplos
- 9.4.1. Historia de HTML5
- 9.4.2. Elementos de HTML5
- 9.4.3. APIS
- 9.4.4. CCS3

9.5. Modelo de Objeto de Documento
- 9.5.1. ¿Qué es el Modelo de Objetos del Documento?
- 9.5.2. Uso de DOCTYPE
- 9.5.3. La importancia de validar el HTML
- 9.5.4. Accediendo a los elementos
- 9.5.5. Creando elementos y textos
- 9.5.6. Usando innerHTML
- 9.5.7. Eliminando un elemento o nodo de texto
- 9.5.8. Lectura y escritura de los atributos de un elemento
- 9.5.9. Manipulando los estilos de los elementos
- 9.5.10. Adjuntar múltiples ficheros a la vez

9.6. Introducción a CSS y ejemplos
- 9.6.1. Sintaxis CSS3
- 9.6.2. Hojas de estilo
- 9.6.3. Etiquetas
- 9.6.4. Selectores
- 9.6.5. Diseño web con CSS

9.7. Introducción a JavaScript y ejemplos
- 9.7.1. ¿Qué es JavaScript?
- 9.7.2. Breve historia del lenguaje
- 9.7.3. Versiones de JavaScript
- 9.7.4. Mostrar un cuadro de diálogo
- 9.7.5. Sintaxis de JavaScript
- 9.7.6. Comprensión de Scripts
- 9.7.7. Espacios
- 9.7.8. Comentarios
- 9.7.9. Funciones
- 9.7.10. JavaScript en la página y externo

9.8. Funciones en JavaScript
- 9.8.1. Declaraciones de función
- 9.8.2. Expresiones de función
- 9.8.3. Llamar a funciones
- 9.8.4. Recursividad
- 9.8.5. Funciones anidadas y cierres
- 9.8.6. Preservación de variables
- 9.8.7. Funciones multi-anidadas
- 9.8.8. Conflictos de nombres
- 9.8.9. Clausuras o cierres
- 9.8.10. Parámetros de una función

9.9. PlayCanvas para desarrollar juegos web
- 9.9.1. ¿Qué es PlayCanvas?
- 9.9.2. Configuración del proyecto
- 9.9.3. Creando un objeto
- 9.9.4. Agregando físicas
- 9.9.5. Añadiendo un modelo
- 9.9.6. Cambiando los ajustes de gravedad y escena
- 9.9.7. Ejecutando Scripts
- 9.9.8. Controles de cámara

9.10. Phaser para desarrollar juegos web
- 9.10.1. ¿Qué es Phaser?
- 9.10.2. Cargando recursos
- 9.10.3. Construyendo el mundo
- 9.10.4. Las plataformas
- 9.10.5. El jugador
- 9.10.6. Añadir físicas
- 9.10.7. Usar el teclado
- 9.10.8. Recoger *Pickups*
- 9.10.9. Puntos y puntuación
- 9.10.10. Bombas de rebote

Módulo 10. Redes y sistemas multijugador

10.1. Historia y evolución de videojuegos multijugador
- 10.1.1. Década 1970: primeros juegos multijugador
- 10.1.2. Años 90: Duke Nukem, Doom, Quake
- 10.1.3. Auge de videojuegos multijugador
- 10.1.4. Multijugador local y online
- 10.1.5. Juegos de fiesta

10.2. Modelos de negocio multijugador
- 10.2.1. Origen y funcionamiento de los modelos de negocio emergentes
- 10.2.2. Servicios de venta en línea
- 10.2.3. Libre para jugar
- 10.2.4. Micropagos
- 10.2.5. Publicidad
- 10.2.6. Suscripción con pagos mensuales
- 10.2.7. Pagar por juego
- 10.2.8. Prueba antes de comprar

10.3. Juegos locales y juegos en red
- 10.3.1. Juegos locales: inicios
- 10.3.2. Juegos de fiesta: Nintendo y la unión de la familia
- 10.3.3. Juegos en red: inicios
- 10.3.4. Evolución de los juegos en red

10.4. Modelo OSI: Capas I
- 10.4.1. Modelo OSI: introducción
- 10.4.2. Capa física
- 10.4.3. Capa de enlace de datos
- 10.4.4. Capa de red

10.5. Modelo OSI: Capas II
- 10.5.1. Capa de transporte
- 10.5.2. Capa de sesión
- 10.5.3. Capa de presentación
- 10.5.4. Capa de aplicación

10.6. Redes de computadores e internet
- 10.6.1. ¿Qué es una red de computadoras?
- 10.6.2. Software
- 10.6.3. Hardware
- 10.6.4. Servidores
- 10.6.5. Almacenamiento en red
- 10.6.6. Protocolos de red

10.7. Redes móviles e inalámbricas
- 10.7.1. Red móvil
- 10.7.2. Red inalámbrica
- 10.7.3. Funcionamiento de las redes móviles
- 10.7.4. Tecnología digital

10.8. Seguridad
- 10.8.1. Seguridad personal
- 10.8.2. *Hacks y cheats* en videojuegos
- 10.8.3. Seguridad anti-trampas
- 10.8.4. Análisis de sistemas de seguridad anti-trampas

10.9. Sistemas multijugador: servidores
- 10.9.1. Alojamiento de servidores
- 10.9.2. Videojuegos MMO
- 10.9.3. Servidores de videojuegos dedicados
- 10.9.4. LAN Parties

10.10. Diseño de videojuegos multijugador y programación
- 10.10.1. Fundamentos de diseño de videojuegos multijugador en Unreal
- 10.10.2. Fundamentos de diseño de videojuegos multijugador en Unity
- 10.10.3. Como hacer que un juego multijugador sea divertido
- 10.10.4. Más allá de un mando: innovación en controles multijugador

05

Metodología

Este programa de capacitación ofrece una forma diferente de aprender. Nuestra metodología se desarrolla a través de un modo de aprendizaje de forma cíclica: ***el Relearning***.

Este sistema de enseñanza es utilizado, por ejemplo, en las facultades de medicina más prestigiosas del mundo y se ha considerado uno de los más eficaces por publicaciones de gran relevancia como el ***New England Journal of Medicine.***

Descubre el Relearning, un sistema que abandona el aprendizaje lineal convencional para llevarte a través de sistemas cíclicos de enseñanza: una forma de aprender que ha demostrado su enorme eficacia, especialmente en las materias que requieren memorización"

Estudio de Caso para contextualizar todo el contenido

Nuestro programa ofrece un método revolucionario de desarrollo de habilidades y conocimientos. Nuestro objetivo es afianzar competencias en un contexto cambiante, competitivo y de alta exigencia.

Con TECH podrás experimentar una forma de aprender que está moviendo los cimientos de las universidades tradicionales de todo el mundo"

Accederás a un sistema de aprendizaje basado en la reiteración, con una enseñanza natural y progresiva a lo largo de todo el temario.

El alumno aprenderá, mediante actividades colaborativas y casos reales, la resolución de situaciones complejas en entornos empresariales reales.

Un método de aprendizaje innovador y diferente

El presente programa de TECH es una enseñanza intensiva, creada desde 0, que propone los retos y decisiones más exigentes en este campo, ya sea en el ámbito nacional o internacional. Gracias a esta metodología se impulsa el crecimiento personal y profesional, dando un paso decisivo para conseguir el éxito. El método del caso, técnica que sienta las bases de este contenido, garantiza que se sigue la realidad económica, social y profesional más vigente.

> *Nuestro programa te prepara para afrontar nuevos retos en entornos inciertos y lograr el éxito en tu carrera"*

El método del caso ha sido el sistema de aprendizaje más utilizado por las mejores escuelas de negocios del mundo desde que éstas existen. Desarrollado en 1912 para que los estudiantes de Derecho no solo aprendiesen las leyes a base de contenidos teóricos, el método del caso consistió en presentarles situaciones complejas reales para que tomasen decisiones y emitiesen juicios de valor fundamentados sobre cómo resolverlas. En 1924 se estableció como método estándar de enseñanza en Harvard.

Ante una determinada situación, ¿qué debería hacer un profesional? Esta es la pregunta a la que te enfrentamos en el método del caso, un método de aprendizaje orientado a la acción. A lo largo de 4 años, te enfrentarás a múltiples casos reales. Deberás integrar todos tus conocimientos, investigar, argumentar y defender tus ideas y decisiones.

Relearning Methodology

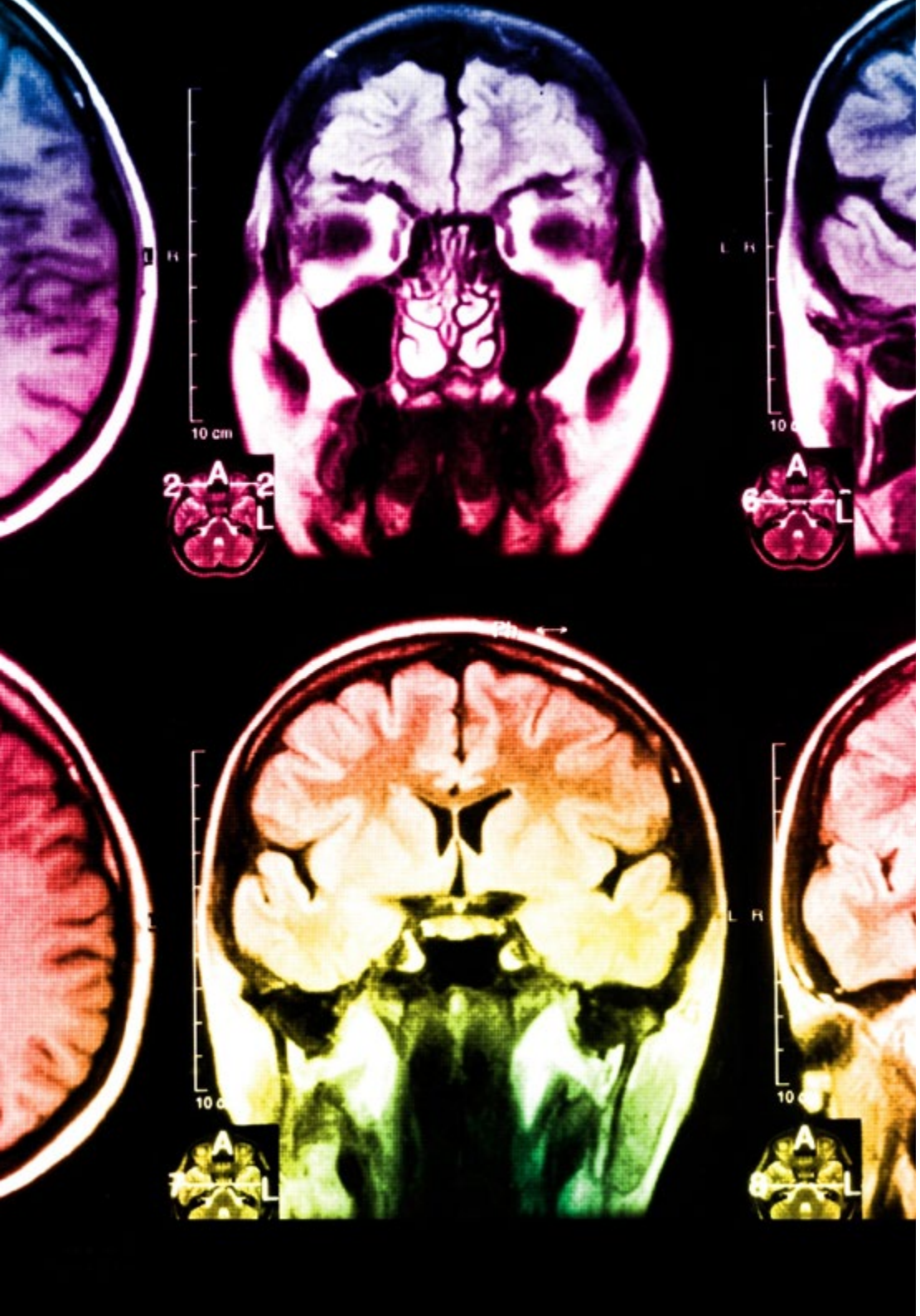
Nuestra Universidad es la primera en el mundo que combina los case studies de Harvard University con un sistema de aprendizaje 100 % online basado en la reiteración, que combina elementos didácticos diferentes en cada lección.

Potenciamos los case studies de Harvard con el mejor método de enseñanza 100 % online: el Relearning.

En 2019 obtuvimos los mejores resultados de aprendizaje de todas las universidades online en español en el mundo.

En TECH aprenderás con una metodología vanguardista concebida para capacitar a los directivos del futuro. Este método, a la vanguardia pedagógica mundial, se denomina Relearning.

Nuestra Universidad es la única en habla hispana licenciada para emplear este exitoso método. En 2019 conseguimos mejorar los niveles de satisfacción global de nuestros alumnos (calidad docente, calidad de los materiales, estructura del curso, objetivos…) con respecto a los indicadores de la mejor universidad online en español.



En nuestro programa el aprendizaje no es un proceso lineal, sino que sucede en espiral (aprender, desaprender, olvidar y reaprender). Por eso, se combinan cada uno de estos elementos de forma concéntrica. Con esta metodología se han capacitado más de 650.000 graduados universitarios con un éxito sin precedentes. En ámbitos tan distintos como la bioquímica, la genética, la cirugía, el derecho internacional, las habilidades directivas, las ciencias del deporte, la filosofía, el derecho, la ingeniería, el periodismo, la historia o los mercados e instrumentos financieros. Todo ello en un entorno de alta exigencia, con un alumnado universitario de un perfil socioeconómico alto y una media de edad de 43,5 años.

El relearning te permitirá aprender con menos esfuerzo y más rendimiento, implicándote más en tu capacitación, desarrollando el espíritu crítico, la defensa de argumentos y el contraste de opiniones: una ecuación directa al éxito.

A partir de la última evidencia científica en el ámbito de la neurociencia, no solo sabemos organizar la información, las ideas, las imágenes, los recuerdos, sino que sabemos que el lugar y el contexto donde hemos aprendido algo es fundamental para que seamos capaces de recordarlo y almacenarlo en el hipocampo, para retenerlo en nuestra memoria a largo plazo.

De esta manera, y en lo que se denomina Neurocognitive context-dependent e-learning, los diferentes elementos de nuestro programa están conectados con el contexto donde el participante desarrolla su práctica profesional.

Este programa ofrece los mejores materiales educativos, preparados a conciencia para los profesionales:

Material de estudio

Todos los contenidos didácticos son creados por los especialistas que van a impartir el curso, específicamente para él, de manera que el desarrollo didáctico sea realmente específico y concreto.

Estos contenidos son aplicados después al formato audiovisual, para crear el método de trabajo online de TECH. Todo ello, con las técnicas más novedosas que ofrecen piezas de gran calidad en todos y cada uno los materiales que se ponen a disposición del alumno.

Clases magistrales

Existe evidencia científica sobre la utilidad de la observación de terceros expertos.

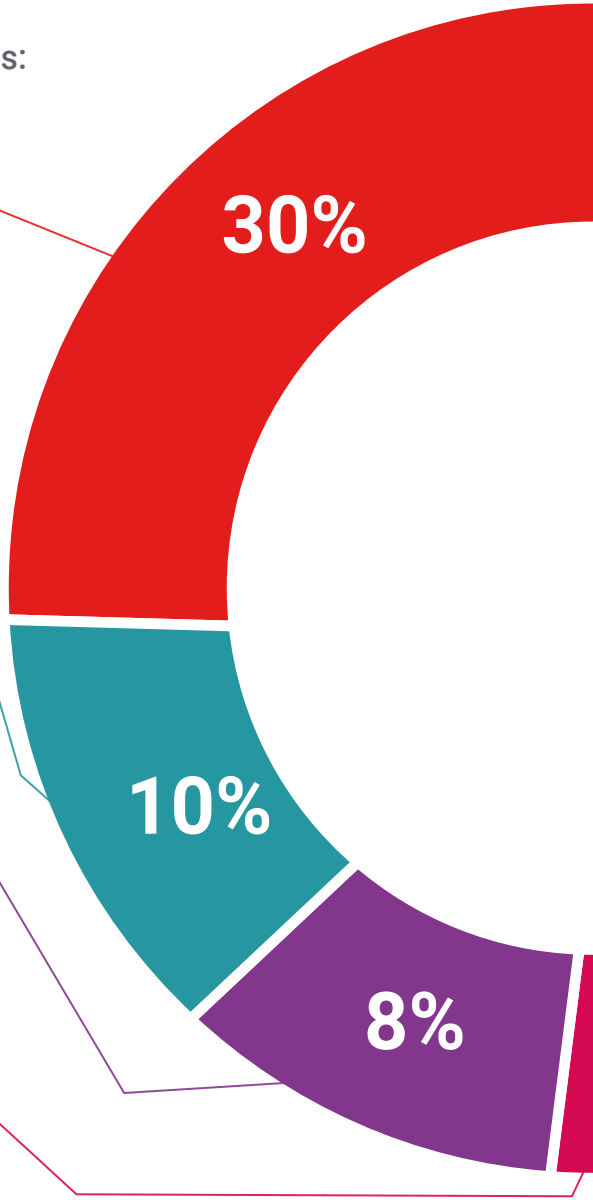
El denominado Learning from an Expert afianza el conocimiento y el recuerdo, y genera seguridad en las futuras decisiones difíciles.

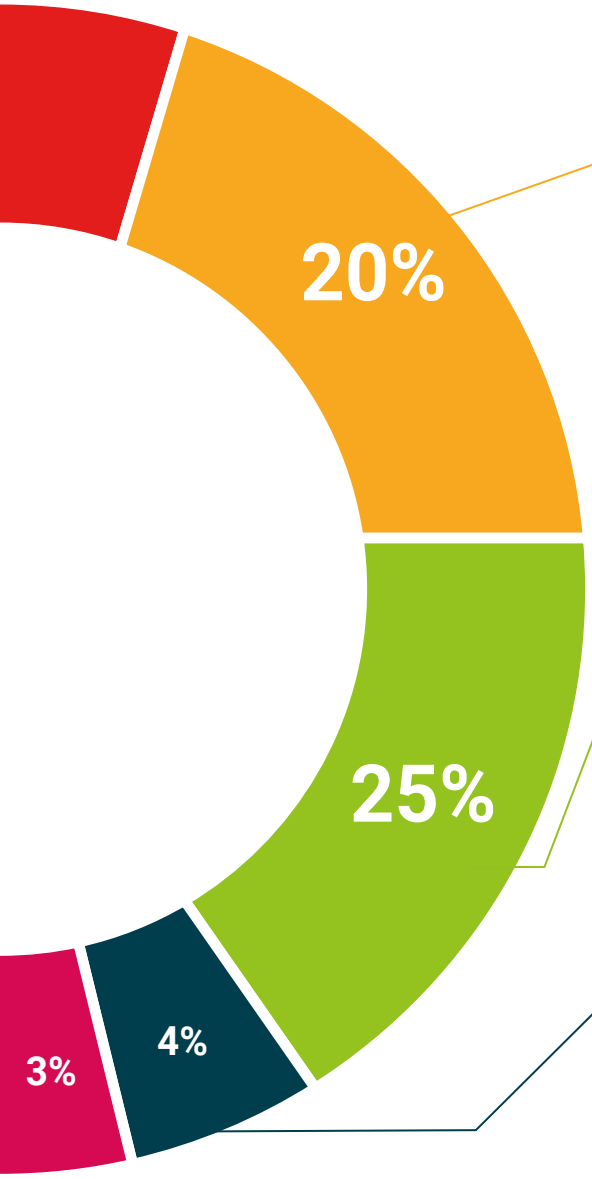
Prácticas de habilidades y competencias

Realizarán actividades de desarrollo de competencias y habilidades específicas en cada área temática. Prácticas y dinámicas para adquirir y desarrollar las destrezas y habilidades que un especialista precisa desarrollar en el marco de la globalización que vivimos.

Lecturas complementarias

Artículos recientes, documentos de consenso y guías internacionales, entre otros. En la biblioteca virtual de TECH el estudiante tendrá acceso a todo lo que necesita para completar su capacitación.





Case Studies

Completarán una selección de los mejores cases studies de la materia que se emplean en Harvard. Casos presentados, analizados y tutorizados por los mejores especialistas del panorama internacional.

Resúmenes interactivos

El equipo de TECH presenta los contenidos de manera atractiva y dinámica en píldoras multimedia que incluyen audios, vídeos, imágenes, esquemas y mapas conceptuales con el fin de afianzar el conocimiento.

Este exclusivo sistema educativo para la presentación de contenidos multimedia fue premiado por Microsoft como "Caso de éxito en Europa".

Testing & Retesting

Se evalúan y reevalúan periódicamente los conocimientos del alumno a lo largo del programa, mediante actividades y ejercicios evaluativos y autoevaluativos: para que, de esta manera, el estudiante compruebe cómo va consiguiendo sus metas.

06

Titulación

El Máster Título Propio en Programación de Videojuegos garantiza, además de la capacitación más rigurosa y actualizada, el acceso a dos diplomas de Máster Propio, uno expedido por TECH Global University y otro expedido por la Universidad Latinoamericana y del Caribe.

Supera con éxito este programa y recibe tu titulación universitaria sin desplazamientos ni farragosos trámites”

El programa del **Máster Título Propio en Programación de Videojuegos** es el más completo del panorama académico actual. A su egreso, el estudiante recibirá un diploma universitario emitido por TECH Global University, y otro por la Universidad Latinoamericana y del Caribe.

Estos títulos de formación permanente y actualización profesional de TECH Global University y Universidad Latinoamericana y del Caribe garantizan la adquisición de competencias en el área de conocimiento, otorgando un alto valor curricular al estudiante que supere las evaluaciones y acredite el programa tras cursarlo en su totalidad.

Este doble reconocimiento, de dos destacadas instituciones universitarias, suponen una doble recompensa a una formación integral y de calidad, asegurando que el estudiante obtenga una certificación reconocida tanto a nivel nacional como internacional. Este mérito académico le posicionará como un profesional altamente capacitado y preparado para enfrentar los retos y demandas en su área profesional.

Título: **Máster Título Propio en Programación de Videojuegos**

Modalidad: **online**

Duración: **12 meses**

Acreditación: **60 ECTS**

tech global university

D/Dña _____, con documento de identificación _____ ha superado con éxito y obtenido el título de:

Máster Título Propio en Programación de Videojuegos

Se trata de un título propio de 1.500 horas de duración equivalente a 60 ECTS, con fecha de inicio dd/mm/aaaa y fecha de finalización dd/mm/aaaa.

TECH Global University es una universidad reconocida oficialmente por el Gobierno de Andorra el 31 de enero de 2024, que pertenece al Espacio Europeo de Educación Superior (EEES).

En Andorra la Vella, a 28 de febrero de 2024

Dr. Pedro Navarro Illana
Rector

código único TECH: AFWOR23S techtitute.com/titulos

UNIVERSIDAD LATINOAMERICANA Y DEL CARIBE

La integración por el conocimiento

ULAC

La Vida Integral

CERTIFICA A:

_____

Identificación: _____

Por haber superado con éxito y obtenido el título de:

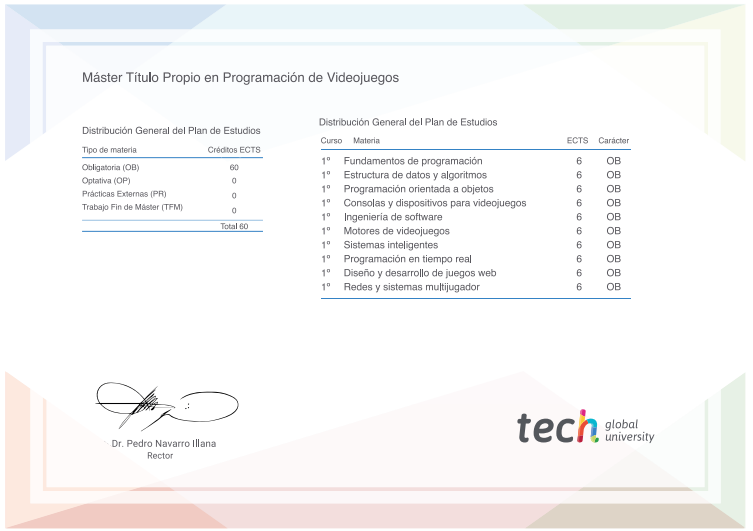
MÁSTER TÍTULO PROPIO EN PROGRAMACIÓN DE VIDEOJUEGOS

En la ciudad de Caracas a los 04 días del mes de diciembre de 2023

Dr. Elmo Bell-Smythe
Rector de la ULAC

Rafa Aramayo
CEO de La Vida Integral

Verifique la validez y autenticidad de este Certificado escaneando el código QR



Máster Título Propio en Programación de Videojuegos

Distribución General del Plan de Estudios

Tipo de materia	Créditos ECTS
Obligatoria (OB)	60
Optativa (OP)	0
Prácticas Externas (PR)	0
Trabajo Fin de Máster (TFM)	0
	Total 60

Distribución General del Plan de Estudios

Curso	Materia	ECTS	Carácter
1º	Fundamentos de programación	6	OB
1º	Estructura de datos y algoritmos	6	OB
1º	Programación orientada a objetos	6	OB
1º	Consolas y dispositivos para videojuegos	6	OB
1º	Ingeniería de software	6	OB
1º	Motores de videojuegos	6	OB
1º	Sistemas inteligentes	6	OB
1º	Programación en tiempo real	6	OB
1º	Diseño y desarrollo de juegos web	6	OB
1º	Redes y sistemas multijugador	6	OB

Dr. Pedro Navarro Illana
Rector

tech global university

*Apostilla de La Haya. En caso de que el alumno solicite que su título en papel recabe la Apostilla de La Haya, TECH Universidad ULAC realizará las gestiones oportunas para su obtención, con un coste adicional.

tech universidad ULAC

Máster Título Propio
Programación de Videojuegos

- Modalidad: **online**
- Duración: **12 meses**
- Titulación: **TECH Universidad ULAC**
- Acreditación: **60 ECTS**
- Horario: **a tu ritmo**
- Exámenes: **online**

Máster Título Propio
Programación de Videojuegos

tech universidad ULAC